NASA-CR-200981

FINAL TECHNICAL REPORT

Project: Preliminary C++ Class Definitions for Object-Oriented Engine Simulation

Period covered: October 1993 - January 1995

PI: Bruce K. Walker
Assoc. Prof. of Aerospace Engineering and Engineering Mechanics
Univ. of Cincinnati
Cincinnati, OH 45221-0070

Phone: 513-556-3552
FAX: 513-556-4589
Email: bwalker@uceng.uc.edu or bruce.walker@uc.edu

Submitted to: NASA Lewis Research Center
Interdisciplinary Technology Office
M.S. 142-5
21000 Brookpark Rd.
Cleveland, OH 44135

Department of Aerospace Engineering and Engineering Mechanics
University of Cincinnati

May, 1996

FINAL TECHNICAL REPORT

Preliminary C++ Class Definitions for Object-Oriented Engine Simulation

Background

The objective of this work was to define and code some preliminary object classes in C++ for eventual use in the National Propulsion System Simulator (NPSS) program at NASA Lewis Research Center (LeRC). The class definition phase of the project involved collaboration with LeRC researchers and with researchers from General Electric Aircraft Engines (GEAE) on the class hierarchy and the selection and definition of the classes that were needed. The coding phase involved the writing, compilation, and testing of C++ code to implement some of the preliminary classes, with interface requirements specified primarily by GEAE. The original goal was to define and code classes for the implementation of simple functional tasks like table lookups for gas properties and interfacing with a variable table class that was under construction at LeRC and to implement some simple components such as ducts and bends. This was to lead to a follow-on project that involved implementing in the new modular C++ structure a compressor rig simulation, where the compressor simulation to be mimicked was to be ether one that had previously been implemented in FORTRAN by GEAE or the NEPP compressor that had been examined by LeRC. The motivation there was the a compressor rig simulation involves essentailly all of the complexity that would be required for simulating any other engine component and therefore that other components could mimic the structure for a compressor rig simulation. However, funding for the follow-on project was not available.

Progress Summary

The project began in November of 1993 with numerous meetings among the UC research team and GEAE to establish guidelines for the classes needed and how to handle the transmission of data among the various classes. These discussions about and revisions to the class hierarchy continued for several months into the project.

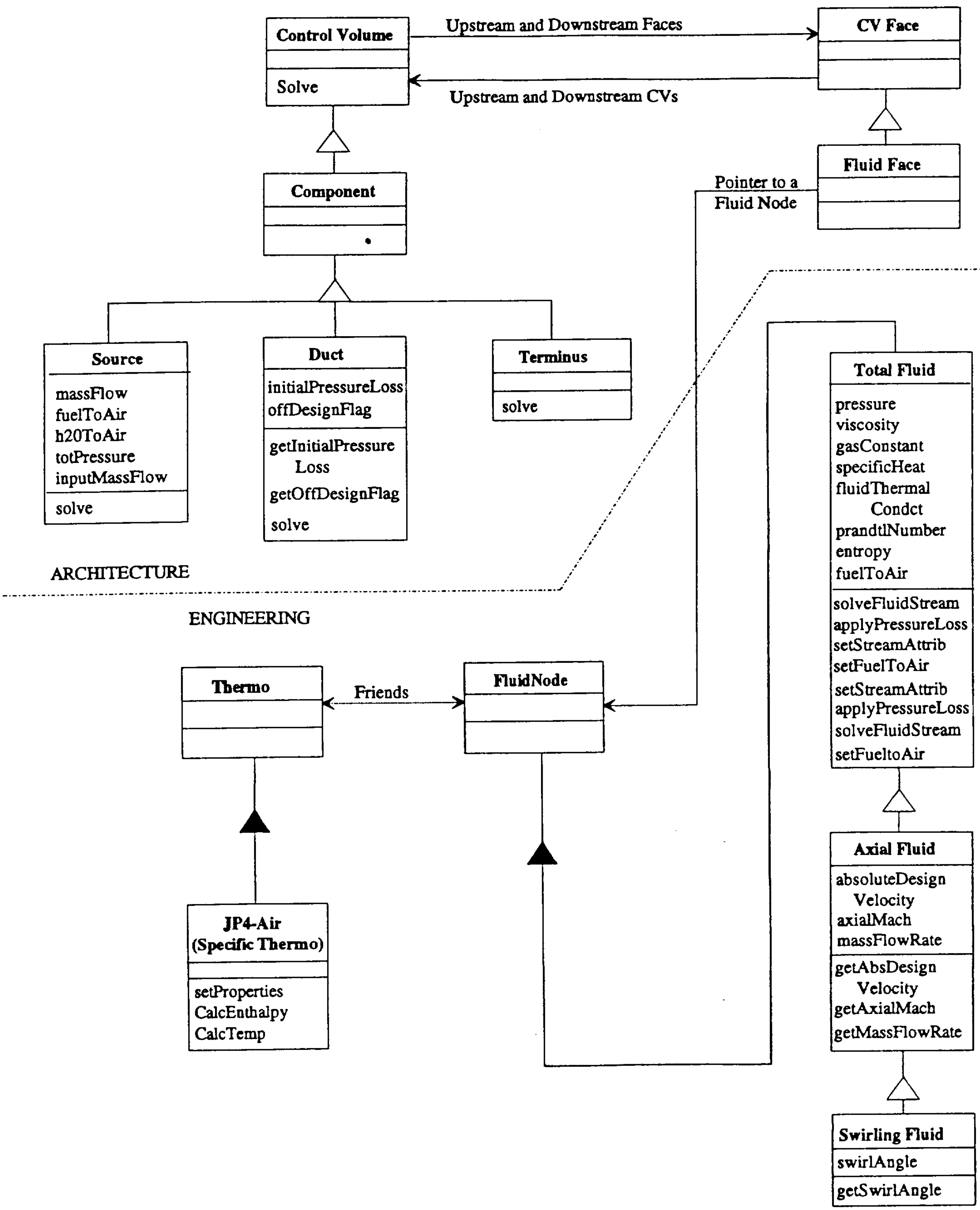
The decision was made at the early meetings to focus initially on building the architecture and classes necessary to simulate a simple fluid duct section carrying a homogeneous fluid.

Coincident to this work, several versions of the class hierarchy were developed and discussed with GEAE and with LeRC researchers. Much effort was expended on establishing the appropriate dependencies and data handling protocols for the classes. In constructing our class hierarchy, we attempted to maintain a structure similar to one suggested by NPSS researchers at NASA LeRC.

A major task in the class hierarchy design was the definition of a 'connector' class to connect classes representing adjacent physical flow components, which we ultimately chose to model after the then-current 'CVFace' class that the NPSS project was using. Also, the decision on where the fluid variables (pressure, temperature, etc.) should reside was discussed at length and ultimately resulted in the decision to place these variables in the separate 'Fluid' classes that define the fluid properties (as opposed to within the component class itself).

One unknown factor throughout the class hierarchy discussions involved the need to interface data with the solvers that would ultimately control the calculations needed to first initialize and then to run a transient simulation. Our goal was to maintain compatibility with any symbol table class concept that might be decided upon as a NPSS standard.

Class Hierarchy



The class hierarchy that we eventually decided upon is illustrated in Figure 1. noted, however, that this was not the first class hierarchy concept for which we began writing code. In fact, complete code was written for a slightly different class architecture that successfully simulated steady duct flow, but then this code was scrapped when the new structure illustrated by Figure 1 was developed. The new architecture eliminated a redundant class that had been present in the previous architecture to represent control volume faces. The functions and data that were associated with these elements were moved to the "Fluid Face" classes. Also, the resident classes of some of the variables were changed to reflect the need for the corresponding values at higher levels of the overall hierarchy. Figure 1 indicates the resident variables and member functions for each class for which the UC research group wrote code.

From GEAE, we obtained FORTRAN code for GEAE's "fluid handler", "fluid node", and "stream starter" components. Respectively, these implement essentially the fluid thermodynamic properties, the interface, and the source for a general fluid flow component. We also obtained code representing the three types of nondynamic (i.e. steady flow) duct cases usually considered by GEAE in their propulsion system simulators, namely axial, total and swirling duct components. Some of this code was FORTRAN, some C-like pseudocode. We translated all of these codes to C++.

For C++ coding rules, we chose to use the Ellemtel rules that were in effect at the time of the project. For source code control, we used RCS on the IBM RS6000 workstation cluster at UC. NASA LeRC personnel and GEAE personnel were given accounts on the UC cluster so that they had easy access to all code that was developed.

The first version of a one-dimensional, nondynamic duct section code was completed in final form in February, 1994. However, shortly thereafter (March, 1994), the expected architecture for the NPSS modules changed following discussions among NPSS personnel at LeRC, our GEAE collaborators, and other industry personnel. This effectively negated our original duct section class hierarchy, so we began then to change the duct section classes to reflect the new architecture.

New versions of the duct section classes code were completed in the early summer of 1994. These classes were compiled and tested under Borland C++ Version 4.0 for PCs and under the xlC C++ compiler for IBM RS6000 workstations. They were also compiled using Gnu G++ Version 2.4.3 for IBM RS6000 workstations, but the compiled code yielded an execution time error indicating that an input file was not present even though it was. This error remained unresolved.

The summer of 1994 represented a hiatus in the project. Both students supported on the grant were funded by other sources (one went to NASA LeRC as a summer intern in the NPSS program) while the PI for the grant devoted only minimal time to the effort. As a result, a no-cost extension of the grant was sought to extend the funding through January of 1995.

By the time that substantial work on the project resumed in the autumn of 1994, the NPSS group at NASA LeRC had made further chagnes to the class hierarchy, essentially replacing the 'Control Volume' and 'CV Face' classes in Figure 1 with a more general 'Gas Flux' class that allowed for separate accounting for total and static conditions of the fluid. This new class structure rendered the duct codes from the early summer invalid again, but the final version of the NPSS class structure remained in some doubt, so no effort was made to rewrite the duct simulation code.

Meanwhile, technical discussions with GEAE about the compressor rig simulation had begun over the summer, and preliminary efforts began in September, 1994, to add the classes necessary to implement the compressor rig simulation. FORTRAN code for the GEAE compressor rig components was obtained from GEAE in October. The bulk of the work performed from then until the termination of the project involved the translation of the compressor rig physics into C++ code using the class hierarchy established earlier in the project. This task was not fully complete when the project funding terminated.

Additionally, in September of 1994, several discussions were held with GEAE on the appropriate technical tasks to be proposed to NASA LeRC for renewal of the project. The final proposal to NASA LeRC, which was sent in late October, 1994, proposed the development of C++ based controller classes for the compressor rig simulator using the transient simulation solver that NASA LeRC was then in the preliminary stages of developing. This effort was to be essentially a C++ version of the compressor rig controller development that GEAE was doing internally using FORTRAN routines. Unfortunately, funding was not available to support this effort.

Personnel

The PI throughout the effort was Prof. Bruce K. Walker of the UC Department of Aerospace Engineering and Engineering Mechanics. Technical assistance was provided by Prof. Prem K. Khosla of the UC Department of Aerospace Engineering and Engineering Mechanics. The students supported by the grant and their period of support were: Kenneth T. Moore (November, 1993 - June, 1994), Jason W. Mitchell (November, 1993 - June, 1994, and October, 1994 - November, 1994), and Yousef Mannaa (November, 1994 - January, 1995), all of whom were graduate students in the UC Department of Aerospace Engineering and Engineering Mechanics.

The primary GEAE contact for the project was Frank D. Parsons, with occasional input from Benny Anderson.

The NASA LeRC technical monitor was Greg Follen. Technical contacts at NASA LeRC for this project included Brian Curlett, Jack Gould, James Felder, and Gretchen Davidian.

In addition to numerous email and phone contacts, face-to-face meetings on this project were held with NPSS personnel at NASA LeRC on 3 December 1993, 19 January 1994, 11 March 1994, and 18 October 1994, and at UC on 20 April 1994. Meetings with Frank Parsons of GEAE were held on a regular basis, usually biweekly, throughout the project period with numerous email and phone contacts in between.